Software Quality Management

2004-2005

Marco Scotto (Marco.Scotto@unibz.it)
Contents

- Definitions
- Quality of the software product
- Special features of software
- Early software quality models
  - Boehm model
  - McCall model
- Standard ISO 9126
Software: intellectual product consisting of information stored on a storage device (ISO/DIS 9000: 2000)

- Software may occur as concepts, transactions, procedures. One example of software is a computer program.
- Software is "intellectual creation comprising the programs, procedures, rules and any associated documentation pertaining to the operation of a data processing system."
- A **software product** is the "complete set of computer programs, procedures and associated documentation and data designated for delivery to a user" [ISO 9000-3].
- Software is independent of the medium on which it is recorded.
Quality of the software product

➢ The product should, on the highest level...
   • Ensure the satisfaction of the user needs
   • Ensure its proper use

➢ Earlier: 1 developer, 1 user
   • The program should run and produce results similar to those expected

➢ Later: more developers, more users
   • Need to economical use of the storage devices
   • Understandability, portability
   • User-friendliness, learnability

➢ Nowadays:
   • Efficiency, reliability, no errors, able to restart without using data
Why is software "different"?

- Does not really have "physical" existence
- It is developed within a special life cycle
- It changes very rapidly
- It always has to be adaptable to the newest hardware and software environment
- The customers faces difficulties when specifying the requirements
- The customer has extremely high expectations regarding software
- Defining the quality attributes of software is extremely difficult
- Measuring the value of the quality attributes is extremely difficult
- Copying, packaging is not problematic
Early software quality models (2/6)

- In the 1970’s: mostly theoretical models
  - Knuth: proving correctness of small programs using mathematical approach

- 1975-1980: first software quality models
  - Define the elements of software quality and their interconnections (tables of correlation, priorities, etc.)
  - “Fixed” quality models
  - “Self defined” quality models
Early software quality models (3/6)

- **The Boehm model**
  - 1977, Barry Boehm
    (http://sunset.usc.edu/Research_Group/barry.html)
  - a TRW Systems and Energy Inc. (“Thompson Ramo Wooldridge”) software research lab

- **McCall model**
  - 1978, James McCall
  - Project manager at General Electric
Both models concentrate on the final product

The models define
- Basic user needs
- Quality factors of the product (high level quality attributes)
- Quality factors are decomposed into quality criteria
- The quality attributes have associated metrics
Early software quality models (5/6)

We have to suppose that any software user is interested in the following questions when purchasing a new software:

- Is the product usable in its present state?
- How easily can the product be changed?
- How will it be possible to transfer the product into an other hardware / software environment?
According to the hierarchy reflected by the questions, the two models make use of the following three basic principles in classifying a product:

1. Usability of the product in its present state (in the Boehm model: "as is utility", in the McCall model: "product operation")
2. Changeability of the product’s functions (in the Boehm model: "maintainability", in the McCall model: "product revision")
3. Portability of the product into an other hardware-software environment (in the Boehm model: "portability", in the McCall model: "product transition")
Early software quality models

- **Boehm**
  - Basic user requirements (3): “as is” utility, general usability, maintainability
  - Quality factors (7): portability, reliability, efficiency, usability, testability, understandability, flexibility
  - Quality characteristics (12)

- **McCall**
  - Basic user requirements (3): product operation, product revision, product transition
  - Quality characteristics (11): usability, integrity, efficiency, correctness, reliability, maintainability, testability, flexibility, reusability, portability, interoperability
  - Quality characteristics (22)
The product-oriented approach
## Tables of correlation

<table>
<thead>
<tr>
<th></th>
<th>Correctness</th>
<th>Efficiency</th>
<th>Integrity</th>
<th>Ease of use</th>
<th>Maintainability</th>
<th>Testability</th>
<th>Changeability</th>
<th>Portability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correctness</td>
<td>-</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Efficiency</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Integrity</td>
<td>-</td>
<td>●</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ease of use</td>
<td>x</td>
<td>●</td>
<td>x</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Maintainability</td>
<td>x</td>
<td>●</td>
<td>x</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>-</td>
</tr>
<tr>
<td>Testability</td>
<td>x</td>
<td>●</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>-</td>
</tr>
<tr>
<td>Changeability</td>
<td>x</td>
<td>●</td>
<td>-</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Portability</td>
<td>●</td>
<td>-</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

x = positive correlation

● = negative correlation
Correlation of the quality factors (1/2)

- Many correlations weaken the attributes, therefore the weight of different quality factors should be different
- Prioritizing of the quality factors
Correlation of the quality factors (2/2)

- **Boehm model**
  - Maintainability is dealt with in very much detail

- **McCall model**
  - Precise measurement of “as is” utility
Boehm and McCall models

- Fixed quality models
- There are “define it by yourself” quality models
  - The attributes are defined together with the user
ISO 9126 standard (1/2)

- Software specialists have looked for a standard that would unambiguously define the quality attributes of the software
- The standard was based on:
  - McCall and Boehm models
  - Experience in using the models
  - Existing business needs
ISO 9126 standard (2/2)

- Requirements for the quality characteristics in the new of standard:
  - To cover together all aspects of software quality resulting from ISO quality definition
  - To describe the product quality with a minimum of overlap
  - To be as close as possible to the established technology
  - To form a set of not more than six to eight characteristics for reason of clarity and handling
  - To identify areas of attributes of software products for further refinement
## Software quality characteristics in ISO 9126

<table>
<thead>
<tr>
<th>Quality characteristics</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>A set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy the stated or implied needs.</td>
</tr>
<tr>
<td>Reliability</td>
<td>A set of attributes that bear on the capability of the software to maintain its level of performance under stated conditions for a stated period of time.</td>
</tr>
<tr>
<td>Usability</td>
<td>A set of attributes that bear on the effort needed to use, and on the individual assessment of such use, by a stated or implied set of users.</td>
</tr>
<tr>
<td>Efficiency</td>
<td>A set of attributes that bear on the relationship between the level of performance of the software and the amount of resources used, under stated conditions.</td>
</tr>
<tr>
<td>Maintainability</td>
<td>A set of attributes that bear on the effort needed to make specified modifications.</td>
</tr>
<tr>
<td>Portability</td>
<td>A set of attributes that bear on the ability of software to be transferred from one environment to another.</td>
</tr>
</tbody>
</table>
## Sub-characteristics (1/5)

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Sub-characteristics</th>
<th>Definition of sub-characteristic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality</td>
<td>Suitability</td>
<td>Attribute of software that bears on the presence and appropriateness of a set of functions for specified tasks.</td>
</tr>
<tr>
<td></td>
<td>Accuracy</td>
<td>Attributes of software that bear on the provision of right or agreed results or effects.</td>
</tr>
<tr>
<td></td>
<td>Interoperability</td>
<td>Attributes of software that bear on its ability to interact with specified systems.</td>
</tr>
<tr>
<td></td>
<td>Compliance</td>
<td>Attributes of software that make the software adhere to application related standards or conventions or regulations in laws and similar prescriptions.</td>
</tr>
<tr>
<td></td>
<td>Security</td>
<td>Attributes of software that bear on its ability to prevent unauthorized access, whether accidental or deliberate, to programs and data.</td>
</tr>
</tbody>
</table>
### Sub-characteristics (2/5)

<table>
<thead>
<tr>
<th>Reliability</th>
<th>Maturity</th>
<th>Attributes of software that bear on the frequency of failure by faults in the software.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fault tolerance</td>
<td></td>
<td>Attributes of software that bear on its ability to maintain specified level of performance in cases of software faults or of infringement of its special interface.</td>
</tr>
<tr>
<td>Recoverability</td>
<td></td>
<td>Attributes of software that bear on the capability to re/establish its level of performance and recover the data directly affected in case of a failure and on the time and effort needed for it.</td>
</tr>
</tbody>
</table>
### Sub-characteristics (3/5)

<table>
<thead>
<tr>
<th>Usability</th>
<th>Understandability</th>
<th>Attributes of software that bear on the users’ effort for recognizing the logical concept and its applicability.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Learnability</td>
<td>Attributes of software that bear on the users’ effort for learning its application (for example, operation control, input, output).</td>
</tr>
<tr>
<td></td>
<td>Operability</td>
<td>Attributes of software that bear on the users’ effort for operation and operation control.</td>
</tr>
</tbody>
</table>
## Sub-characteristics (4/5)

<table>
<thead>
<tr>
<th>Sub-characteristic</th>
<th>Attribute (4/5)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Efficiency</strong></td>
<td>Time behavior</td>
<td>Attributes of software that bear on response and processing times and throughput rates in performing its function.</td>
</tr>
<tr>
<td></td>
<td>Resource behavior</td>
<td>Attributes of software that bear on the amount of resources used and the duration of such use in performing its function.</td>
</tr>
<tr>
<td><strong>Maintainability</strong></td>
<td>Analyzability</td>
<td>Attributes of software that bear on the effort needed for diagnosis of deficiencies or causes of failures, or for identification of parts to be modified.</td>
</tr>
<tr>
<td></td>
<td>Changeability</td>
<td>Attributes of software that bear on the effort needed for modification, fault removal or for environmental change.</td>
</tr>
<tr>
<td></td>
<td>Stability</td>
<td>Attributes of software that bear on the risk of unexpected effect of modifications.</td>
</tr>
<tr>
<td></td>
<td>Testability</td>
<td>Attributes of software that bear on the effort needed for validating the modified software.</td>
</tr>
</tbody>
</table>
### Sub-characteristics (5/5)

<table>
<thead>
<tr>
<th>Portability</th>
<th>Adaptability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attributes of software that bear on the opportunity for its adaptation to different specified environments without applying other actions or means than those provided for this purpose for the software considered.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Installability</th>
<th>Attributes of software that bear on the effort needed to install the software in a specified environment.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Conformance</th>
<th>Attributes of software that make the software adhere to standards or conventions relating to portability.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Replace ability</th>
<th>Attributes of software that bear on the opportunity and effort of using it in the place of specified other software in the environment of that software.</th>
</tr>
</thead>
</table>
Developing ISO 9126 further

Developing ISO 9126 further

- Standard ISO/IEC 9126-1.2 introduces the concept of *Quality in use*
  - As being "the user's view of the quality of an environment containing software, and is measured from the results of using the software in the environment, rather than properties of the software itself"
  - The quality in the user's environment may be different from that in the developer's environment, because some functions may not be visible to a user or may not be used by a user
## Subcharacteristics of quality in use

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effectiveness</td>
<td>The capability of the software product to enable users to achieve specified goals with accuracy and completeness in a specified context of use.</td>
</tr>
<tr>
<td>Productivity</td>
<td>The capability of the software product to enable users to expend appropriate amounts of resources in relation to the effectiveness achieved in a specified context of use.</td>
</tr>
<tr>
<td>Safety</td>
<td>The capability of the software product to achieve acceptable levels of risk of harm to people, software, equipment or the environment in a specified context of use.</td>
</tr>
<tr>
<td>Satisfaction</td>
<td>The capability of the software product to satisfy users in a specified context of use.</td>
</tr>
</tbody>
</table>
ISO 9126

quality in use
  - effectiveness
  - productivity
  - safety
  - satisfaction

functionality
  - accuracy
  - suitability
  - interoperability
  - compliance
  - security
  - maturity

reliability
  - fault tolerance
  - recoverability
  - availability

usability
  - understandability
  - learnability
  - operability

efficiency
  - time behaviour
  - resource utilisation

maintainability
  - analysability
  - changeability
  - stability
  - testability

portability
  - adaptability
  - installability
  - co-existence
  - conformance
  - replaceability

Software Quality Management
The product-oriented approach

- **Quality characteristics**
  - Examples
    - usability
    - maintainability
    - portability
    - time behaviour
    - etc.

- **Measures/design actions**
  - Examples
    - implementing help messages
    - modular design
    - parameterization of programs
    - etc.

Software Quality Management
A matrix for linking measures/design actions to quality characteristics

- **Effectiveness**
  - Modeling
  - Controlling
  - Optimalisation
  - Styling
  - Tracing
  - Strengthening

- **Reliability**
  - Efficiency
  - Ergonomics
  - Changeability
  - Robustness
  - Portability

The matrix indicates the relationships between different actions and quality characteristics with '+' for improvement and '-' for degradation.
Choosing the appropriate quality attributes

- Some general principles / findings / correlations
- Principles / findings / correlations according to the particular case
A double loop for specification and realization of quality characteristics

- Quality needs
- Quality characteristics
- Software attributes
- Measures/design actions
- Business system characteristics

User/customer is central
Developer is central
The Quality House for the determination of quality characteristics

- relations between characteristics
- user/customer needs
- relations between needs and quality characteristics
- prioritised needs
- prioritised quality characteristics
- quality characteristics

Software Quality Management
Selection of relevant quality characteristics

- Business process
- Software product
- User/customer

Quality characteristics
Translation process
Quality profile
Quality profiles - examples

<table>
<thead>
<tr>
<th>Application/environment characteristics</th>
<th>Software quality characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human lives affected</td>
<td>Integrity</td>
</tr>
<tr>
<td></td>
<td>Reliability</td>
</tr>
<tr>
<td></td>
<td>Correctness</td>
</tr>
<tr>
<td></td>
<td>Verifiability</td>
</tr>
<tr>
<td>Longlife cycle</td>
<td>Maintainability</td>
</tr>
<tr>
<td></td>
<td>Expandability</td>
</tr>
<tr>
<td>Experimental system or high rate of change</td>
<td>Flexibility</td>
</tr>
<tr>
<td>Experimental technology in hardware design</td>
<td>Portability</td>
</tr>
<tr>
<td>Many changes over life cycle</td>
<td>Flexibility</td>
</tr>
<tr>
<td></td>
<td>Reusability</td>
</tr>
<tr>
<td></td>
<td>Expandability</td>
</tr>
<tr>
<td>Real time application</td>
<td>Efficiency</td>
</tr>
<tr>
<td></td>
<td>Reliability</td>
</tr>
<tr>
<td></td>
<td>Correctness</td>
</tr>
<tr>
<td>On-board computer application</td>
<td>Efficiency</td>
</tr>
<tr>
<td></td>
<td>Reliability</td>
</tr>
<tr>
<td></td>
<td>Correctness</td>
</tr>
<tr>
<td>Processing of classified information</td>
<td>Integrity</td>
</tr>
<tr>
<td>Interrelated systems</td>
<td>Interoperability</td>
</tr>
</tbody>
</table>
Business analysis

- **Business process**
  - importance
  - risks
  - control

- User
  - knowledge level
  - number
  - age

- Software product
  - type of product
  - type of usage
  - infrastructure
Investigating business process characteristics

• Importance for the organization
• Risk for the environment/organization
• Control
  • Complexity
  • Dynamics
  • Stability
Example

- Continuous critical management decisions
- Strong dependency of the organization from software products
  - Leads to high requirements for reliability (availability)
Reliability (1/2)

- The capability of the software to maintain the level of performance of the system when used under specified conditions

- **Maturity:**
  - The capability of the software to avoid failure as a result of faults in the software
  - **Example**
    - *Mean Time Between Failures* = Operation time divided by number of failures

- **Fault tolerance:**
  - The capability of the software to maintain a specified level of performance in cases of software faults or of infringement of its specified interface
  - **Example**
    - *Input Error Detection Ratio* = Number of detected input errors versus total number of inputs
Reliability (2/2)

- **Recoverability:**
  - The capability of the software to re-establish its level of performance and recover the data directly affected in the case of a failure
  - **Example:**
    - *Mean Down Time* = The total down time of a system divided by the number of observed breakdowns

- **Availability**
  - the capability of the software to be in a state to perform a required function at a given point in time, under stated conditions of use
  - **Example:**
    - *Relative Availability Percentage* = The ratio of time the software product is available to the time it is needed
Example

• Many business process parameters
• Many interdependencies
• High level of difficulty regarding business control and monitoring

 Leads to specific requirements for understandability (of the software application)
Investigating user characteristics

- business process
  - importance
  - risks
  - control
- **user**
  - *knowledge level*
  - *number*
  - *age*
- software product
  - type of product
  - type of usage
  - infrastructure
Investigating user characteristics

• Type (end users, intermediate, remote)
• Knowledge level (educational background)
• Number
• Age
• Experience (business process, automation)
Examples

• Many inexperienced users
• Relatively low educational background
  ➢ Leads to high requirements for learnability

• Many managers as users
• Relatively high experience
  ➢ Leads to high requirements for response time
  (time behavior)
Usability (1/2)

• The capability of the software to be understood, learned, used and liked by the user, when used under specified conditions

➢ **Understandability:**

• The capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use

• **Example:**

  ▪ **Understandable Input and Output Data Items Ratio =**
  
  Number of Functions for which an average new user understands input/output data items related to total amount of functions
Usability (2/2)

- **Learnability:**
  - The capability of the software product to enable the user to learn its application
  - **Example:**
    - *Average Learning Time* = The average time necessary for a set of inexperienced users to achieve a specific level of competence with the system

- **Operability:**
  - The capability of the software product to enable the user to operate and control it
  - **Example:**
    - *Human Error Free Time* = Total operation time by a group of inexperienced users divided by the number of human errors
Efficiency (1/2)

- The capability of the software to provide the required performance relative to the amount of resources used, under stated conditions.

- **Time behaviour:**
  - the capability of the software to provide appropriate response and processing time and throughput rates when performing its function, under stated conditions.
  - **Example**
    - *Response Time* = Average time between (end of a) command and the moment of gaining a result.
Efficiency (2/2)

- **Resource utilization**
  - The capability of the software to use appropriate resources in an appropriate time when the software performs its function under stated conditions

- **Example**
  - \[ ROM Utilization Distribution per Function = \frac{\text{Number of bytes in ROM necessary for each function / feature}}{\text{ROM space used for more than one function should be divided according to function specific ROM-usage}} \]
Investigating software product characteristics (1/2)

• business process
  • importance
  • risks
  • control
• user
  • knowledge level
  • number
  • Age

• software product
  • type of product
  • type of usage
  • infrastructure
Investigating the software product characteristics (2/2)

- **Type of system:**
  - functionality
  - language used
  - business application/embedded software

- **Type of usage:**
  - on-line/batch
  - intensity/frequency
  - time criticality

- **Type of infrastructure:**
  - type of hardware
  - number of locations
  - network
  - communication
Example

- Many batch-jobs
- Transaction processing system (operational processes)
  - Leads to high requirements for recoverability
Example of a quality profile result

For each selected characteristic:

- selection of subcharacteristics
- importance rating (evaluation level)
- context description (business situation)
- “Standard” ways to specify and assess on each level (metrics)

<table>
<thead>
<tr>
<th>ISO 9126 quality characteristics</th>
<th>Evaluation level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-</td>
</tr>
<tr>
<td>FUNCTIONALITY</td>
<td></td>
</tr>
<tr>
<td>RELIABILITY</td>
<td>✓</td>
</tr>
<tr>
<td>USABILITY</td>
<td>✓</td>
</tr>
<tr>
<td>EFFICIENCY</td>
<td>X</td>
</tr>
<tr>
<td>MAINTAINABILITY</td>
<td>✓</td>
</tr>
<tr>
<td>PORTABILITY</td>
<td>X</td>
</tr>
</tbody>
</table>
## Assignment of levels of evaluation

<table>
<thead>
<tr>
<th>levels</th>
<th>safety view</th>
<th>economy view</th>
<th>security view</th>
<th>environmental view</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>many people killed</td>
<td>financial disaster (company will not survive)</td>
<td>protection of strategic data and services</td>
<td>unrecoverable environmental damage</td>
</tr>
<tr>
<td>B</td>
<td>threat to human lives</td>
<td>large economic loss (company engaged)</td>
<td>protection of critical data and services</td>
<td>local pollution</td>
</tr>
<tr>
<td>C</td>
<td>damage to property, few people injured</td>
<td>protection against error risk</td>
<td>no environmental risk</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>small damage to property, no risk to people</td>
<td>no specific risk identified</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Identification and specification of quality characteristics

Five steps:

1. Investigating in a structured way with users/customers the business characteristics
2. Determination of the characteristics of the business processes
3. Determination of the characteristics of the software product
4. Determination of the characteristics of users
5. Specification of quality characteristics from the users point of view
Summary

Metric

Quality attribute

Definition

Product  Process  Resource